|  |
| --- |
|  |
| **Encryption Project** |
| Documentation |

|  |
| --- |
| Kyle Jones  12-9-2015 |

# **Table of Contents**

[**Table of Contents** 1](#_Toc447631905)

[**Introduction** 3](#_Toc447631906)

[Overview 3](#_Toc447631907)

[Objectives 3](#_Toc447631908)

[**Planning** 4](#_Toc447631909)

[Requirements 4](#_Toc447631910)

[Trello 4](#_Toc447631911)

[**Research** 6](#_Toc447631912)

[Cryptography 6](#_Toc447631913)

[Substitution Ciphers 6](#_Toc447631914)

[Monoalphabetic Substitution Ciphers 6](#_Toc447631915)

[Atbash Cipher 6](#_Toc447631916)

[Caesar Cipher 6](#_Toc447631917)

[Polyalphabetic Substitution Ciphers 6](#_Toc447631918)

[Vigenere Cipher 6](#_Toc447631919)

[Autokey Cipher 6](#_Toc447631920)

[Running Key Cipher 7](#_Toc447631921)

[One-Time Pad 7](#_Toc447631922)

[Mechanical Substitution Ciphers 7](#_Toc447631923)

[Enigma 7](#_Toc447631924)

[Chaocipher 7](#_Toc447631925)

[SIGABA (ECM Mark II) 7](#_Toc447631926)

[Typex (BID/08) 8](#_Toc447631927)

[Transposition Ciphers 8](#_Toc447631928)

[Rail Fence Cipher 8](#_Toc447631929)

[Route Cipher 8](#_Toc447631930)

[Columnar Transposition Cipher 8](#_Toc447631931)

[Double Transposition Cipher 8](#_Toc447631932)

[Myszkowski Transposition Cipher 8](#_Toc447631933)

[Hashing Encryption 9](#_Toc447631934)

[SHA-1 9](#_Toc447631935)

[Block Ciphers 9](#_Toc447631936)

[DES 9](#_Toc447631937)

[AES 9](#_Toc447631938)

[Stream Ciphers 9](#_Toc447631939)

[Vernam Cipher 9](#_Toc447631940)

[RC4 9](#_Toc447631941)

[Methods of Testing 9](#_Toc447631942)

[Code Reviews 9](#_Toc447631943)

[Performance Analysis 9](#_Toc447631944)

[Benchmarking 10](#_Toc447631945)

[Specification Based Testing 10](#_Toc447631946)

[Usability Testing 10](#_Toc447631947)

[**Development** 11](#_Toc447631948)

[Design 11](#_Toc447631949)

[UML 11](#_Toc447631950)

[Structural Diagrams 11](#_Toc447631951)

[Class Diagram 11](#_Toc447631952)

[Behavioral Diagrams 11](#_Toc447631953)

[Activity Diagram 11](#_Toc447631954)

[Sequence Diagram 11](#_Toc447631955)

[Pseudocode 11](#_Toc447631956)

[Software and Tools 15](#_Toc447631957)

[CoderPad 15](#_Toc447631958)

[Visual Studio 15](#_Toc447631959)

[Microsoft Word 15](#_Toc447631960)

[Git 15](#_Toc447631961)

[Construction 15](#_Toc447631962)

[Testing 16](#_Toc447631963)

[Test Strategy 16](#_Toc447631964)

[Test Plan 16](#_Toc447631965)

[Optimizations 16](#_Toc447631966)

[**Evaluation** 17](#_Toc447631967)

[**Appendices** 18](#_Toc447631968)

[Appendix A 18](#_Toc447631969)

[Appendix B 18](#_Toc447631970)

[**References** 19](#_Toc447631971)

# **Introduction**

## Overview

Cryptography is defined as “the practice and study of techniques required for secure communication in the presence of third parties.” This topic is the subject of significant research due to its increasing importance in society – particularly with the rising popularity of e-commerce and digitization. This document will describe various forms of cryptography, as well as implementations of some, with an evaluation on each algorithm and the project as a whole.

## Objectives

The objectives for the project are to:

* Gain a better understanding of cryptography and its various methods.
* Achieve a better quality of code.
* Improve project management skills.
* Learn about and incorporate industry standards for testing software.

# **Planning**

## Requirements

The project will have a number of requirements in order to achieve the objectives set out in section <?>. The project will:

* Be documented extensively throughout.
* Adhere to specified targets and timeframes.
* Provide references and diagrams when suitable.

The software will have a number of requirements that will be detailed within this section of the documentation.

As with most pieces of software, the code must be:

* Reliable.
* Efficient.
* Maintainable.
* An appropriate size.

Furthermore, the software must have a set of requirements prior to the start of development which detail the functionality of the software. As such, the software to be created should:

* Have a User Interface (UI) that is easy to use.
* Have the ability to read / write text from / to a text file.
* Encrypt text using various methodologies, based on user input.
* Decrypt text using various methodologies, based on user input.

These requirements will give a number of specific, measurable, attainable, realistic and tangible (SMART) goals to work towards through the course of development. They will also be the basis for the various forms of testing that will be carried out during and after the construction phase of development. All of this information will then contribute towards the evaluation of the overall success of the project.

## Trello

Trello is a collaborative project management tool available online as a website or as an app (Trello, 2011). The tool makes use of ‘boards’, ‘lists’ and ‘cards’. Each board is made up of a number of lists, whereas each list is made up of a number of cards. These help to break down large projects into smaller, manageable tasks in order to provide a realistic goal while giving a better understanding of the project and what is required. Shown below is a screenshot of a Trello board used for this project.

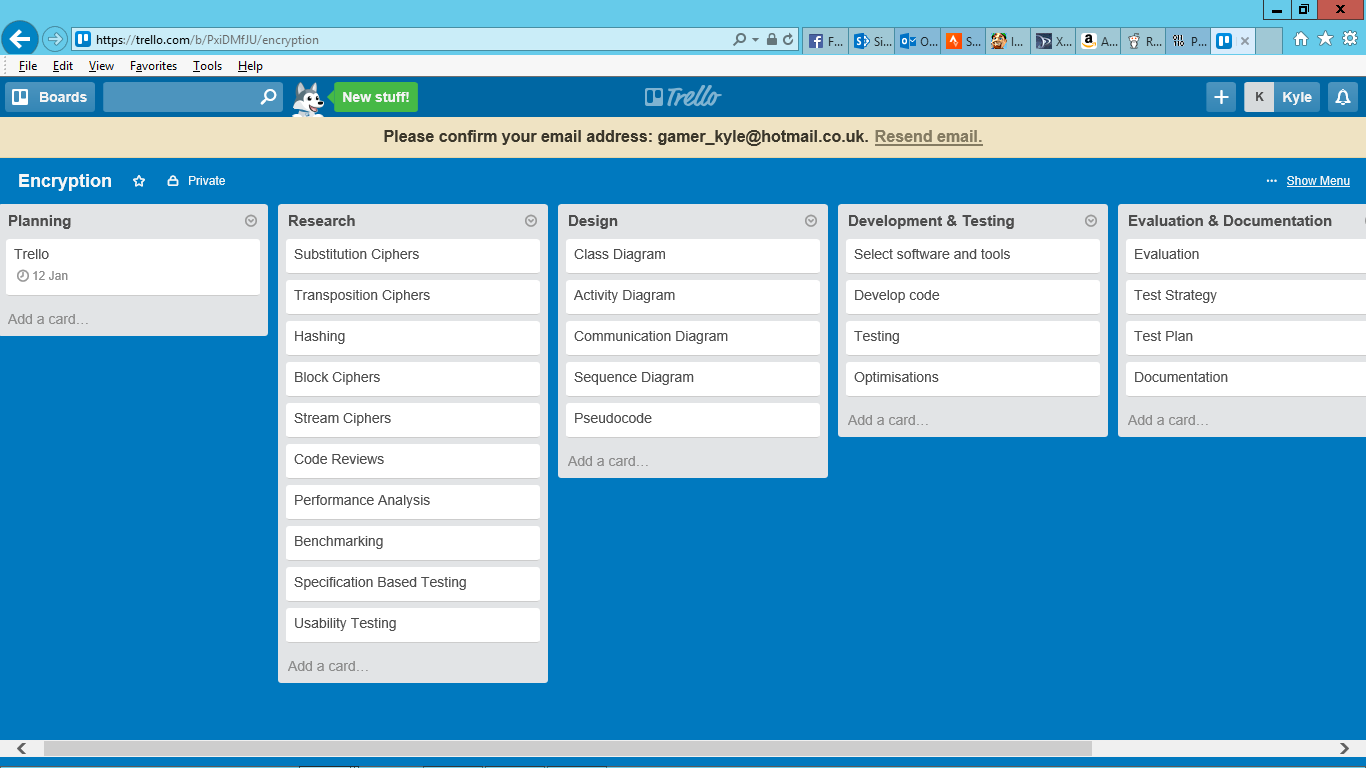


Image Showing Project Trello Board.

# **Research**

## Cryptography

Cryptography is defined as “the process of writing or reading of secret messages or codes” (Merriam-Webster, 2015). Cryptography has been used throughout history in order to prevent messages from being read by individuals other than the intended reader, particularly in times of warfare. The earliest example of cryptography dates back to Ancient Egypt at around 1900BC, with the first cipher being dated back to around 550BC. This was known as the Atbash Cipher and was a simple monoalphabetic cipher used to encode the book of Jeremiah. Since the introduction of cryptography, techniques have advanced significantly in order to prevent cryptanalysis. Cryptanalysis is the process of analyzing and deciphering cryptographic writings or systems (Free Dictionary, 2015).

### Substitution Ciphers

A substitution cipher is the simplest and earliest form of cipher, with the first known instance being the Atbash Cipher used by the Jews circa 550BC. Substitution cipher’s work by taking the characters of the message and substituting them with other characters. There are two forms of substitution cipher – monoalphabetic substitution ciphers and polyalphabetic substitution ciphers. The former simply substitutes the letters using a given position, whereas the polyalphabetic substitution ciphers make use of a secret phrase in order to encode the message through substitution.

#### Monoalphabetic Substitution Ciphers

##### Atbash Cipher

A monoalphabetic substitution cipher where the letters of the alphabet were substituted with the letters found in the same position if the alphabet were reversed. For example, a B was replaced with a Y (\*).

##### Caesar Cipher

Developed by Julius Caesar between the years 50 and 60BC in order to keep military plans secure in transit, the Caesar cipher is one of the most famous ciphers, working by transposing each letter forwards by 3 letters, so that A becomes D etc (\*). Another variation of this cipher is the ROT-13 cipher, where each letter is shifted by 13 letters – half the English alphabet.

#### Polyalphabetic Substitution Ciphers

##### Vigenere Cipher

In 1553, Giovan Belaso proposed the idea of using a password to encrypt messages, an idea that influenced Blaise de Vigenere and eventually evolved into a technique we commonly refer to as an “encryption key”. Vigenere’s cipher is seen as the first encryption key system, transitioning from traditional to modern encryption methods. The cipher is an example of a polyalphabetic substitution cipher and works by taking a key, then using each letter of that key as the number of positions to transpose a letter using a Caesar cipher. For example, with the key “ABC” and the message “XYZ”, the message becomes encrypted to “XZB”.

##### Autokey Cipher

The Autokey cipher was also developed by Blaise de Vigenere by removing the possibility of a pattern search to find the repeating key which could then reveal the key. This is done by taking the standard cipher method proposed by Vigenere but instead adding the new key to the plaintext message.

##### Running Key Cipher

The running key cipher is a variation of the Vigenere cipher which uses a long key, such as an extract from a book, instead of a short key. This means that cryptanalysis is much more difficult as there is no repeating pattern.

#### One-Time Pad

The one-time pad is similar to the running key cipher. However, in this variation, the key is always random. This, combined with the fact that the same key is never used twice, makes it theoretically unbreakable due to every possible decryption is equally likely (Practical Cryptography, 2014).

#### Mechanical Substitution Ciphers

Mechanical Substitution Ciphers are cryptographic tools used to perform complex polyalphabetic substitution encipherment using mechanical rotors. There are a number of notable examples, some of which are described in detail within this section.

##### Enigma

Enigma is the name given to a series of mechanical machines used in World War II to encode and decode messages.

The machine worked using a number of components: a keyboard, a plugboard, a static rotor, a series of three rotors, a reflector and a lampboard (Sale, 2015). The operator presses a letter on the keyboard that they wish to encrypt. This letter then gets passed to the plugboard where it is converted to another letter. This new letter is then passed to the static rotor where it is changed to a new letter before being passed through the other rotors. The other rotors continue to alter the letter until it reaches the reflector, where it is altered once more before being sent back through the rotors and plugboard. After reaching the plugboard, the signal is transmitted to the lampboard, where a letter will be lit to show the final letter.

In Enigma, there were five possible rotors which could be used in 3 slots and with each slot having 26 different starting positions (one for each letter of the English alphabet). This resulted in the machine having roughly 150,000,000,000,000 different combinations of letters (Ellis, 2005). This, coupled with the fact that the Axis powers changed the rotors and their positions daily, a practice similar to that of the one-time pad, meant that Enigma was a very strong method of encryption and is arguably one of the most famous.

##### Chaocipher

(Rubin, 2010).

##### SIGABA (ECM Mark II)

SIGABA was an electric rotor-based cipher machine, making use of three banks of 5 rotors to encode messages (Crypto Museum, 2015). The rotors are split into three unique banks – the Cipher Rotors, the Control Rotors and the Index Rotors. Cipher Rotors connected to Control Rotors, with each rotor having 26 positions on it, each corresponding to a letter of the alphabet. Each of these rotors were identical in production, giving the machine a large advantage of allowing a vast quantity of permutations in setup. This was especially the case as each rotor could be inverted, giving the machine 26! total combinations (Poole, 2013). The Index Rotors on the other hand only had 10 positions each. The unique feature of SIGABA was that most rotor-based crypto machines worked in an odometer function, SIGABA’s Cipher Rotors were altered based on the positions of the Control and Index rotors. The Control Rotors moved the Cipher Rotors, but the Index Rotors altered the function of the Control Rotors. This meant that the Cipher Rotors numbered 1, 2, 3 or 4 would rotate a seemingly random amount, but not all would move. This feature is known as a Stepping Maze.

##### Typex (BID/08)

Typex was the British equivalent of Enigma, which is widely documented as infringing German patents due to it’s overwhelming similarities (Proc, 2012). The machine is built using a series of 5 rotors and a reflector to encode messages. Of the 5 rotors, two were static during encipherment (stators), while the remainder moved in an odometer function (Crypto Museum, 2015).

### Transposition Ciphers

Transposition ciphers work by taking the plaintext message and then re-arranging the letters, so that the message becomes an anagram. The simplest method of doing this is by reversing the message so that it reads backwards.

#### Rail Fence Cipher

The rail fence cipher functions by taking a message and a key in the form of an integer. This key designates the number of lines to be used. The encryption and decryption then works by writing the letters of the message on alternating lines. The ciphertext is then determined by reading left to right, top to bottom – giving the user a scrambled message.

#### Route Cipher

The route cipher is similar to the rail fence cipher, writing the message in columns of a given size and often padding the end with random letters. However, with this variant, the message is encrypted by then writing the letters in a given pattern, such as a clockwise spiral from the top left. This pattern would be pre-arranged with the intended reader so as to ensure that only two individuals could read the plaintext (Braingle, 2014).

#### Columnar Transposition Cipher

Columnar Transposition takes a plaintext message and a numeric key in order to encode the message (Akins, 2008). The plaintext message is written out in a given number of columns. Each column is then given a number which, when combined, makes the numeric key. These columns are then re-arranged into ascending order according to the numeric key, scrambling the message.

#### Double Transposition Cipher

A double transposition is simply a columnar transposition cipher performed twice – once on the plaintext and then once again on the ciphertext received from the first columnar transposition (PBS, 2000).

#### Myszkowski Transposition Cipher

Myszkowski Transposition is a variant of columnar transposition. In regular columnar transposition, a duplicate letter is counted as consecutive numbers, however, in the variation proposed by Myszkowski in 1902 duplicate letters are allocated the same number (Akins, 2008). Columns that have unique numbers are then written downwards whereas those with duplicate numbers are written from left to right.

### Hashing Encryption

#### SHA-1

Secure Hash Algorithm-1 (SHA-1) is a cryptographic hash function used to check that a file has remained unaltered during transmission using a checksum both before and after transmission (AboutTech, 2015). It was developed by the US National Security Agency (NSA) around 1993 and creates a 160-bit hash value known more commonly as a message digest.

### Block Ciphers

A block cipher makes use of a deterministic algorithm that conducts operations on groupings of bits, often in sets of 64 or 128 bits, known as blocks (TechFAQ, 2015). Most modern encryption techniques make use of block ciphers due to them having a broader range of applications and being better analyzed (Stallings, 2010).

#### DES

#### AES

### Stream Ciphers

#### Vernam Cipher

The Vernam cipher is a symmetrical stream cipher developed by Gilbert Vernam, in which a message and a key of the same length have a Boolean “exclusive or” (XOR) function performed on them to give the ciphertext. If a truly random key is used then the cipher can become a One Time Pad, making it unbreakable.

#### RC4

## Methods of Testing

Software testing is a process aimed at evaluating the attributes or capabilities of a piece of software, determining whether it meets the expectations, ensuring quality of the finished product (Pan, 1999). Software testing can be performed in a variety of manners, against many different criteria using a multitude of tools and software to assist in the evaluation. Some of these methods will be the subject of this section of the documentation, and will be used in the testing of the program.

### Code Reviews

A code review is the process of inspecting code with the intention to highlight mistakes to improve upon (Rouse, 2016). Code reviews are a common element in many development processes and methodologies including pair programming and Agile development (Agile Association, 2015).

### Performance Analysis

Examining an application to guarantee that each piece works efficiently is known as performance analysis. The process involves gathering data on the processor usage, storage, input / output and network services. This data then allows the identification of extensive overheads and bottlenecks, allowing the software to be improved using optimization techniques.

### Benchmarking

### Specification Based Testing

Specification Based Testing is broadly defined as the testing of a piece of software against any documented claims concerning the functionality and appearance of the program. (Kaner and Bach, 2005).

### Usability Testing

(ISTQBExamCertification, \*).

# **Development**

## Design

### UML

UML (Unified Modeling Language) is a way of visualizing a piece of software using a number of diagrams (SmartDraw, 2015). There are two groups of UML diagrams – structural diagrams and behavioral diagrams. Structural diagrams are those that define the structure of a piece of software, showing the various objects in a system. Behavioral diagrams on the other hand are those that depict the behavior of a piece of software, explaining how the objects interact with each other (Silva, 2008).

#### Structural Diagrams

##### Class Diagram

Class diagrams are argued to be the most useful UML diagrams. The diagram shows the classes, attributes, operations and relationships in a system. Due to its usefulness, it was decided to design the system by creating a class diagram which can be found below.

#### Behavioral Diagrams

##### Activity Diagram

Activity diagrams are used to illustrate the dynamic nature of a program by modelling the flow of control from one activity to another (Ericsson, 2004). An activity is an operation on a class in the system that results in a change in the system’s state, often classified as a decision.

##### Sequence Diagram

Sequence diagrams are used to illustrate how objects interact and the order in which those interactions occur in a particular scenario ().

### Pseudocode

This section includes the pseudocode created for the software as part of the design, prior to the start of construction of the code. Pseudocode gives a guideline to work from for the construction, fleshing out the design proposed in the UML diagrams with finer details, detailing how the algorithms themselves will look and function (Haas, 2016).

Include iostream;

Include fstream;

Include string;

Using namespace std;

int main()

{

// Run Menu

bool MainMenu();

// End program

return 0;

}

bool MainMenu()

{

// Declare variables

char choice = ‘’;

// Choose method of encryption

Output << “Select a type of encryption to use.” << end line;

Output << “1 – ROT13 << end line << “2 – Vigenere” << end line;

Input >> choice;

Switch(choice)

{

Case 1:

ROT13Encrypt();

Case 2:

Vigenere ();

}

}

ROT13Encrypt()

{

// Declare and initialize variables

String input = “”;

String output = “”;

// User input

Output << "Enter text to encrypt." << end line;

Input >> input;

// Set max length

int max = Length of input;

String output = “”;

// For every letter, apply ROT13 cipher

for (int x = 0; x < max; x++)

{

// Check if character is a letter

if (input[x] isalpha)

{

char letter;

letter = input[x] toUpper);

letter = (((letter - ASCII VALUE OF 'A') + 13) % 26) + ASCII VALUE OF 'A' ;

output += letter;

}

// If not a letter, return character to output

else

{

output += input[x];

}

}

// Output plaintext and encrypted text

Output << "Input: " << input << endl << "Output: " << output;

}

Vigenere()

{

// Declare and initialize variables

string input = "";

string cipher = "";

string output = "";

char selection = ‘’;

char letter = ‘’;

char cipherLetter = ‘’;

int cipherCount = 0;

// User input

Output << “Would you like to encrypt or decrypt?” << end line;

Input >> selection;

If (selection == 1)

{

Output << "Enter a string to encrypt." << end line;

Input >> input;

}

Else

{

Output << "Enter a string to decrypt." << end line;

Input >> input;

}

Output << "Enter a string to use as the cipher." << end line;

Input >> cipher;

// Set max length

int max = Length of input;

int cipherMax = Length of cipher;

// For every letter, apply cipher.

for (int x = 0; x < max; x++)

{

// Used to check if the cipher length is smaller than the total length

if (cipherCount < cipherMax)

{

// Check if character is a letter

if (input[x] isAlpha)

{

// Convert letter to uppercase

letter = input[x] toUpper;

cipherLetter = cipher[cipherCount] toUpper;

If (selection = 1)

{

letter = ((letter + cipherLetter) % 26) + ‘A’;

}

Else

{

letter = ((letter + cipherLetter + 26) % 26) + ‘A’; }

output += letter;

cipherCount++;

}

// If not a letter, return character to output

else

{

output += input[x];

cipherCount++;

}

}

else

{

// Reset cipher position

cipherCount = 0;

// Check if character is a letter

if (input[x] isAlpha)

{

// Convert letters to uppercase

letter = input[x] toUpper;

cipherLetter = cipher[cipherCount] toUpper;

If (selection = 1)

{

letter = ((letter + cipherLetter) % 26) + ‘A’;

}

Else

{

letter = ((letter + cipherLetter + 26) % 26) + ‘A’;

}

output += letter;

cipherCount++;

}

// If not a letter, return character to output

else

{

output += input[x];

cipherCount++;

}

}

}

// Output plaintext and encrypted text

Output << end line << "Input: " << input << end line << "Output: " << output;

}

## Software and Tools

### CoderPad

CoderPad is a real-time collaborative coding environment which can be used to write, compile and run code online (CoderPad, 2015). The website supports the use of C#, C++, Java, Ruby, Python and many more. This makes the website an invaluable tool when coding collaboratively or on hardware that does not have the relevant software installed to carry out this functionality.

### Visual Studio

Visual Studio is an IDE (Integrated Development Environment) developed by Microsoft which includes a large amount of features that are useful for software developers. These features include syntax highlighting, various debugging tools, the ability to export to a range of platforms, Git integration and support for many different languages including C++ and Python (Microsoft, 2015).

### Microsoft Word

Microsoft Word is a piece of word processing software that will be used to create and edit the documentation side of the project. The software comes with a large quantity of features including various formatting options, a dictionary and a thesaurus (Microsoft, 2015).

### Git

Git is a distributed version control system used to manage software projects. Some of the core features provided are bug tracking, branching and merging (Git, 2015). These allow the developer(s) the ability to edit source code without the possibility of overwriting the master version with broken versions during development - providing a consistent, working version. Bug tracking also allows for the identifying and documenting of the development and testing phases of production. Git will serve as the VCS (version control system) for the project, as well as providing a backup and bug tracking.

## Construction

This section details the development of the code used to give the software its functionality. Various techniques and practices have been implemented in the code so as to ensure its maintainability such as the use of comments, indenting and object orientation.

## Testing

Benchmarks, Performance Analysis, Code Review, Usability Testing, Unit Testing

### Test Strategy

See Appendix A.

### Test Plan

See Appendix B.

## Optimizations

# **Evaluation**

# **Appendices**

## Appendix A

## Appendix B

# **References**

AboutTech (2015) “What is SHA-1?”. Available at: <http://pcsupport.about.com/od/termss/g/sha-1.htm> (Accessed: Tuesday 29th March 2016)

Akins, T. (2008) “Myszkowski Transposition”. Available at: <http://candcfamily.comxa.com/ciphers/myszkowski.htm> (Accessed: Monday 1st February 2016)

Akins, T. (2008) “Columnar Transposition”. Available at: <http://rumkin.com/tools/cipher/coltrans.php> (Accessed: Monday 11th January 2016)

Barney, B. (2014) “Performance Analysis Tools”. Available at: <https://computing.llnl.gov/tutorials/performance_tools/#Abstract> (Accessed: Tuesday 2nd February 2016)

Braingle (2014) “Codes and Ciphers: Route Cipher”. Available at: <http://www.braingle.com/brainteasers/codes/route.php> (Accessed: Monday 11th January 2016)

CoderPad (2015) “CoderPad – Interview Your Engineering Candidates Better”. Available at: <https://coderpad.io/> (Accessed: Wednesday 9th December 2015)

Crypto Museum (2015) “Typex (BID/08)”. Available at: <http://www.cryptomuseum.com/crypto/uk/typex/> (Accessed: Monday 1st February 2016)

Crypto Museum (2015) “SIGABA ECM Mark II”. Available at: <http://www.cryptomuseum.com/crypto/usa/sigaba/index.htm> (Accessed: Monday 1st February 2016)

Ellis, C. (2005) “Exploring the Enigma”. Available at: <https://plus.maths.org/content/os/issue34/features/ellis/index> (Accessed: Wednesday 6th January 2016)

Ericsson, M. (2004) “Activity Diagrams: What They Are and How to Use Them”. Available at: <http://www.ibm.com/developerworks/rational/library/2802.html> (Accessed: Tuesday 29th March 2016)

Free Dictionary (2015) “Cryptanalysis – Definition of cryptanalysis by The Free Dictionary”. Available at: <http://www.thefreedictionary.com/cryptanalysis> (Accessed: Tuesday 15th December 2015)

Git (2015) “Git”. Available at: <http://www.git-scm.com/> (Accessed: Wednesday 9th December 2015)

Haas, G. (2016) ‘Pseudocode’ in “Introduction to Computer Programming”. Available at: <http://www.bfoit.org/itp/Pseudocode.html> (Accessed: Tuesday 29th March 2016)

ISTQBExamCertification (\*) “What Is Usability Testing in Software and It’s Benefits to End Users?”. Available at: <http://istqbexamcertification.com/what-is-usability-testing-in-software-and-its-benifits-to-end-user/> (Accessed: Friday 5th February 2016)

Kaner, C., and Bach, J. (2005) “Black Box Software Testing: Specification-Based Testing”. Available at: <http://testingeducation.org/BBST/specbased/BBSTspecBased.pdf> (Accessed: Tuesday 2nd February 2016)

Khan Academy (2015) “Journey into Cryptography”. Available at: <https://www.khanacademy.org/computing/computer-science/cryptography> (Accessed: Friday 18th December 2015)

Merriam-Webster (2015) “Cryptography: Definition of cryptography by Merriam-Webster”. Available at: <http://www.merriam-webster.com/dictionary/cryptography> (Accessed: Tuesday 15th December 2015)

Microsoft (2015) “Microsoft Word – Document & Word Processing Software”. Available at: <https://products.office.com/en-gb/word> (Accessed: Wednesday 9th December 2015)

Microsoft (2015) “Visual Studio – Microsoft Developer Tools”. Available at: <https://www.visualstudio.com/en-us/visual-studio-homepage-vs.aspx> (Accessed: Wednesday 9th December 2015)

MindTools (2015) “Gantt Charts – Planning and Scheduling Team Projects”. Available at: <https://www.mindtools.com/pages/article/newPPM_03.htm> (Accessed: Wednesday 9th December 2015)

Morkel, T. and Eloff, JHP. (2004) “Encryption Techniques: A Timeline Approach”. Available at: <http://icsa.cs.up.ac.za/issa/2004/Proceedings/Research/062.pdf> (Accessed: Tuesday 15th December 2015)

Oxford Dictionaries (2015) “Gantt chart – definition of Gantt chart in English from the Oxford dictionary”. Available at: <http://www.oxforddictionaries.com/definition/english/gantt-chart?q=Gantt+chart> (Accessed: Wednesday 9th December 2015)

Paar, C., and Pelzl, J. (2009) Understanding Cryptography – A Textbook for Students and Practitioners. Springer Science and Business Media.

Pan, J. (1999) “Software Testing”. Available at: <https://users.ece.cmu.edu/~koopman/des_s99/sw_testing/> (Accessed: Monday 1st February 2016)

PBS (2000) “The Double Transposition Cipher”. Available at: <http://www.pbs.org/wgbh/nova/decoding/doubtrans.html> (Accessed: Monday 11th January 2016)

Poole, J. (2013) “SIGABA: ECM Mark II”. Available at: <http://www.usna.edu/Users/math/wdj/_files/documents/sm473-capstone/poole_SIGABA.pdf> (Accessed: Tuesday 2nd February 2016)

Practical Cryptography (2014) “Running Key Cipher”. Available at: <http://www.practicalcryptography.com/ciphers/running-key-cipher> (Accessed: Monday 11th January 2016)

Proc, J. (2012) “Typex”. Available at: [www.jproc.ca/crypto/typex.html](http://www.jproc.ca/crypto/typex.html) (Accessed: Tuesday 2nd February 2016)

Programming Knowledge (2016) “Qt C++ GUI Tutorial”. Available at: <https://www.youtube.com/playlist?list=PLS1QulWo1RIZjrD_OLju84cUaUlLRe5jQ> (Accessed: Tuesday 5th April 2016)

Rodriguez-Clark, D. (2013) “Simple Transposition Ciphers – Crypto Corner”. Available at: <http://crypto.interactive-maths.com/simple-transposition-ciphers.html> (Accessed: Friday 18th December 2015)

Rouse, M. (2016) “What is a code review?”. Available at: <http://searchsoftwarequality.techtarget.com/definition/code-review> (Accessed: Monday 1st February 2016)

Rubin, M. (2010) “Chaocipher Revealed: The Algorithm”. Available at: <http://www.mountainvistasoft.com/chaocipher/ActualChaocipher/Chaocipher-Revealed-Algorithm.pdf> (Accessed: Friday 18th December 2015)

Sale, T. (2015) “The Enigma Cipher Machine”. Available at: <http://www.codesandciphers.org.uk/enigma/index.htm> (Accessed: Friday 18th December 2015)

Silva, N. (2012) “The Complete Guide to UML Diagram Types with Examples”. Available at: <http://creately.com/blog/diagrams/uml-diagram-types-examples/> (Accessed: Tuesday 29th March 2016)

SmartDraw (2015) “UML Diagrams – Learn What They Are and How to Make Them”. Available at: <http://www.smartdraw.com/uml-diagram/> (Accessed: Tuesday 15th December 2015)

Stallings, W. (2010) “Chapter 3 – Block Ciphers and the Data Encryption Standard” in “Cryptography and Network Security: Principles and Practice”. Fifth Edition. Prentice Hall. Available at: <http://www.cs.man.ac.uk/~banach/COMP61411.Info/CourseSlides/Wk2.1.DES.pdf> (Accessed: Friday 5th February 2016)

TechFAQ (2015) “Block and Stream Ciphers”. Available at: <http://www.tech-faq.com/block-and-stream-ciphers.html> (Accessed: Friday 5th February 2016)

TestingBrain (2015) “Software Testing Tutorials”. Available at: <http://www.testingbrain.com/> (Accessed: Friday 18th December 2015)

Trello (2011) “Trello”. Available at: <https://trello.com/> (Accessed: Tuesday 5th April 2016)

Veness, C. (2002) “SHA-1 Cryptographic Hash Algorithm”. Available at: <http://www.movable-type.co.uk/scripts/sha1.html> (Accessed: Tuesday 29th March 2016)